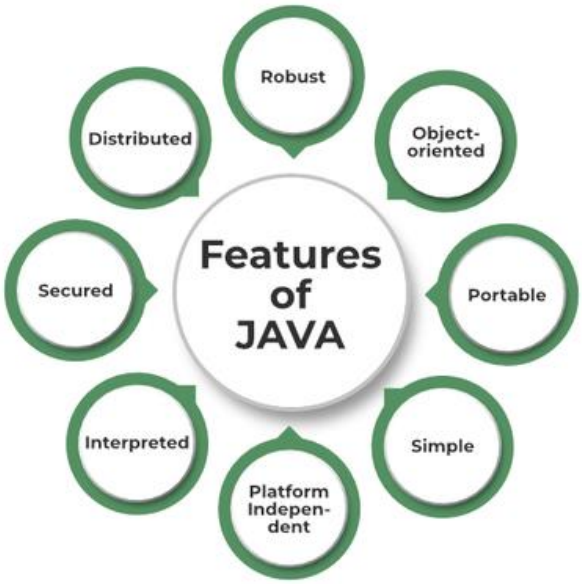
JAVA INTERVIEW QUESTION

1. Is Java Platform Independent if then how?

Yes, Java is a Platform Independent language. Unlike many programming languages javac compiles the program to form a bytecode or .class file. This file is independent of the software or hardware running but needs a JVM(Java Virtual Machine) file preinstalled in the operating system for further execution of the bytecode.

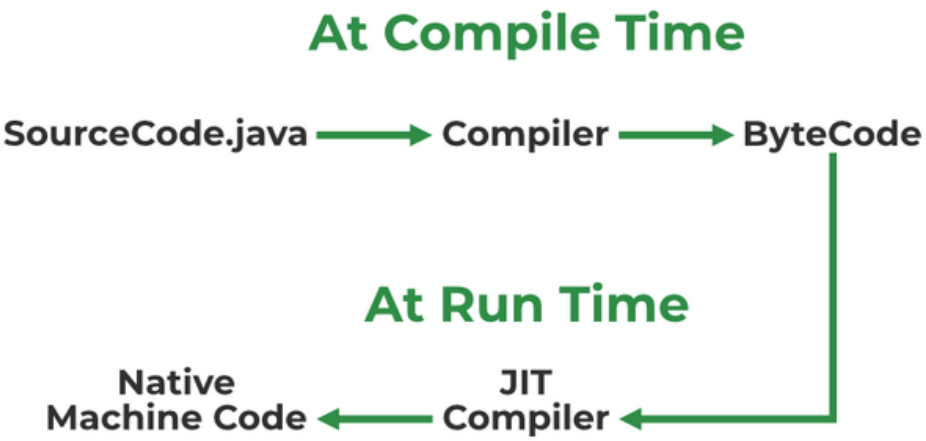
2. What are the top Java Features?



3. What is JVM?

JVM stands for Java Virtual Machine it is a Java interpreter. It is responsible for loading, verifying, and executing the bytecode created in Java.

4. What is JIT?



6. What is a classloader?

Classloader is the part of JRE(Java Runtime Environment), during the execution of the bytecode or created .class file classloader is responsible for dynamically loading the java classes and interfaces to JVM(Java Virtual Machine). Because of classloaders Java run time system does not need to know about files and file systems.

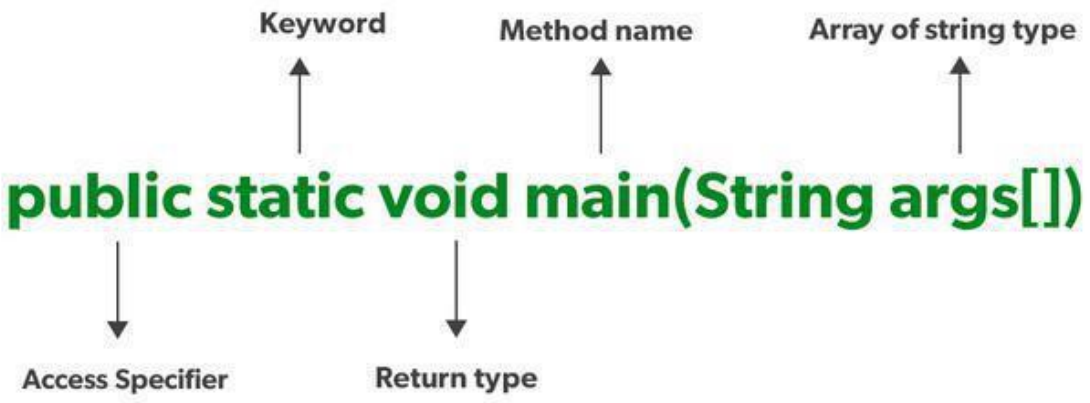
7. Difference between JVM, JRE, and JDK.

JVM: JVM also known as Java Virtual Machine is a part of JRE. JVM is a type of interpreter responsible for converting bytecode into machine-readable code. JVM itself is platform dependent but it interprets the bytecode which is the platform-independent reason why Java is platform-independent.

JRE: JRE stands for Java Runtime Environment, it is an installation package that provides an environment to run the Java program or application on any machine.

JDK: JDK stands for Java Development Kit which provides the environment to develop and execute Java programs. JDK is a package that includes two things Development Tools to provide an environment to develop your Java programs and, JRE to execute Java programs or applications.

9. Explain public static void main(String args[]) in Java.



10. What is Java String Pool?

A Java String Pool is a place in heap memory where all the strings defined in the program are stored. If String is available in the pool, the same object reference is shared with the variable, else a new object is created.

11. What will happen if we don't declare the main as static?

We can declare the main method without using static and without getting any errors. But, the main method will not be treated as the entry point to the application or the program.

11. What will happen if we don't declare the main as static?

We can declare the main method without using static and without getting any errors. But, the main method will not be treated as the entry point to the application or the program.

12. What are Packages in Java?

Packages in Java can be defined as the grouping of related types of classes, interfaces, etc providing access to protection and namespace management.

14. Why Packages are used?

Packages are used in Java in order to prevent naming conflicts, control access, and make searching/locating and usage of classes, interfaces, etc easier.

15. How many types of packages are there in Java?

There are two types of packages in Java

- User-defined packages
- Build In packages

16. Explain different data types in Java.

There are 2 types of data types in Java as mentioned below:

1. Primitive Data Type
2. Non-Primitive Data Type or Object Data type

17. When a byte datatype is used?

A byte is an 8-bit signed two-complement integer. The minimum value supported by bytes is -128 and 127 is the maximum value. It is used in conditions where we need to save memory and the limit of numbers needed is between -128 to 127.

18. Can we declare Pointer in Java?

No, Java doesn't provide the support of Pointer. As Java needed to be more secure because which feature of the pointer is not provided in Java.

19. What is the default value of byte datatype in Java?

The default value of the byte datatype in Java is 0.

21. What is the Wrapper class in Java?Wrapper, in general, is referred to a larger entity that encapsulates a smaller entity. Here in Java, the wrapper class is an object class that encapsulates the primitive data types.

23. Differentiate between instance and local variables.

Instance Variable -Declared outside the method, directly invoked by the method.Has a default value.

Local Variable : Declared within the method No default value

24. What are the default values assigned to variables and instances in Java?

In Java When we haven't initialized the instance variables then the compiler initializes them with default values.

28. What is a static variable?

The static keyword is used to share the same variable or method of a given class. Static variables are the variables that once declared then a single copy of the variable is created and shared among all objects at the class level.

29. What is the difference between System.out, System.err, and System.in?

System.out – It is a PrintStream that is used for writing characters or can be said it can output the data we want to write on the Command Line Interface console/terminal.

System.err – It is used to display error messages.

System.in – It is an InputStream used to read input from the terminal Window. We can't use the System.in directly so we use Scanner class for taking input with the system.in.

30. What do you understand by an IO stream?

36. What is an I/O filter?

An I/O filter also defined as an Input Output filter is an object that reads from one stream and writes data to input and output sources. It used java.io package to use this filter.

38. Difference in the use of print, println, and printf.

print, println, and printf all are used for printing the elements but print prints all the elements and the cursor remains in the same line. println shifts the cursor to next line. And with printf we can use format identifiers too.

39. What are operators?

Operators are the special types of symbols used for performing some operations over variables and values.

41. Explain the difference between >> and >>> operators.

Operators like >> and >>> seem to be the same but act a bit differently. >> operator shifts the sign bits and the >>> operator is used in shifting out the zero-filled bits.

43. What is dot operator?

The Dot operator in Java is used to access the instance variables and methods of class object

45. What is the transient keyword?

The transient keyword is used at the time of serialization if we don't want to save the value of a particular variable in a file.

46. What's the difference between the methods sleep() and wait()?

The sleep() method belongs to the thread class.

Wait() method belongs to the object class.

47. What are the differences between String and StringBuffer?

String	StringBuffer
Store of a sequence of characters.	Provides functionality to work with the strings.
It is immutable.	It is mutable (can be modified and other string operations could be performed on them.)

String	StringBuffer
No thread operations in a string.	It is thread-safe (two threads can't call the methods of StringBuffer simultaneously)

48. What are the differences between StringBuffer and StringBuilder?

StringBuffer	StringBuilder
StringBuffer provides functionality to work with the strings.	StringBuilder is a class used to build a mutable string.
It is thread-safe (two threads can't call the methods of StringBuffer simultaneously)	It is not thread-safe (two threads can call the methods concurrently)
Comparatively slow as it is synchronized.	Being non-synchronized, implementation is faster

49. Which among String Builder or String Buffer should be preferred when there are a lot of updates required to be done in the data?

String is immutable, making it inefficient for scenarios requiring frequent updates. Instead, we can use StringBuilder or StringBuffer. If thread safety is required (synchronized operations), StringBuffer should be used. However, if performance is a priority in a single-threaded context, StringBuilder is the better choice since it is faster and does not incur synchronization overhead.

50. Why is StringBuffer called mutable?

StringBuffer class in Java is used to represent a changeable string of characters. It offers an alternative to the immutable String class by enabling you to change a string's contents without constantly creating new objects.

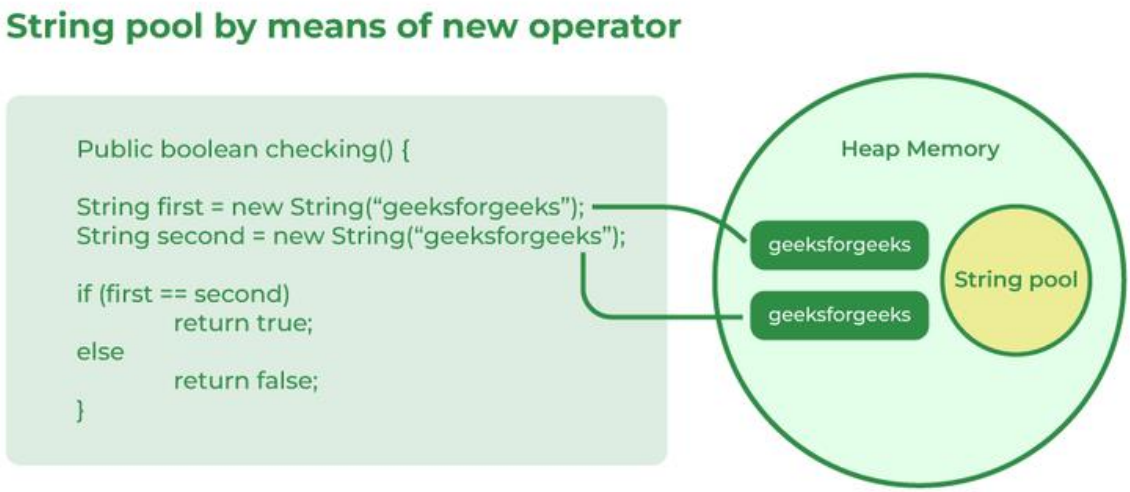
51. How is the creation of a String using new() different from that of a literal?

String using new() is different from the literal as when we declare string it stores the elements inside the stack memory whereas when it is declared using new() it allocates a

dynamic memory in the heap memory. The object gets created in the heap memory even if the same content object is present.

Syntax:

String x = new String("ABC");



52. What is an array in Java?

An Array in Java is a data structure that is used to store a fixed-size sequence of elements of the same type. Elements of an array can be accessed by their index, which starts from 0 and goes up to a length of minus 1.
Syntax:

int[] Arr = new int[5];

53. On which memory arrays are created in Java?

Arrays in Java are created in heap memory. When an array is created with the help of a new keyword, memory is allocated in the heap to store the elements of the array. In Java, the heap memory is managed by the Java Virtual Machine(JVM) and it is also shared between all threads of the Java Program.

54. What are the types of an array?

56. What is the difference between int array[] and int[] array?

int arr[] is a C-Style syntax to declare an Array.

int[] arr is a Java-Style syntax to declare an Array.

57. How to copy an array in Java?

In Java, there are multiple ways to copy an array based on the requirements:

1. clone() Method (Shallow Copy)

58. What do you understand by the jagged array?

A jagged Array in Java is just a two-dimensional array in which each row of the array can have a different length. Since all the rows in a 2-d Array have the same length but a jagged array allows more flexibility in the size of each row.

59. Is it possible to make an array volatile?

In Java, it is not possible to make a volatile. Volatile keywords in Java can only be applied to individual variables but not to arrays or collections.

60 . What are the main concepts of OOPs in Java?

The main concepts of OOPs in Java are mentioned below:

- Inheritance
- Polymorphism
- Abstraction
- Encapsulation

61. How is the 'new' operator different from the 'newInstance()' operator in Java?

the new operator is used to create objects, but if we want to decide the type of object to be created at runtime, there is no way we can use the new operator. In this case, we have to use the newInstance() method.

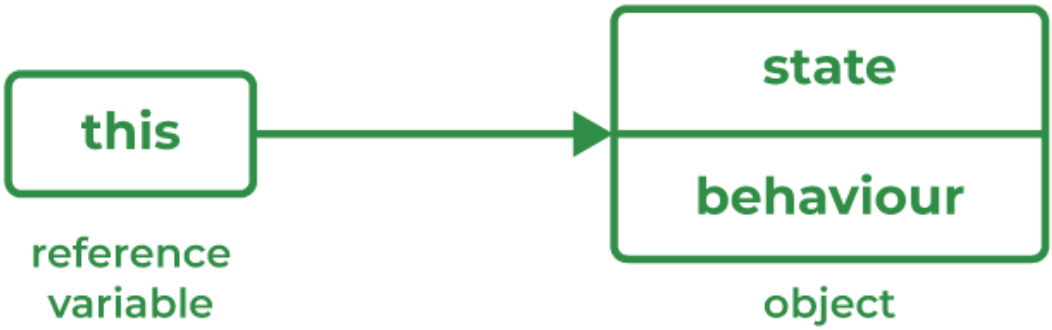
62. What are Classes in Java?

In Java, Classes are the collection of objects sharing similar characteristics and attributes. Classes represent the blueprint or template from which objects are created. Classes are not real-world entities but help us to create objects which are real-world entities.

63. What is the difference between static (class) method and instance method?

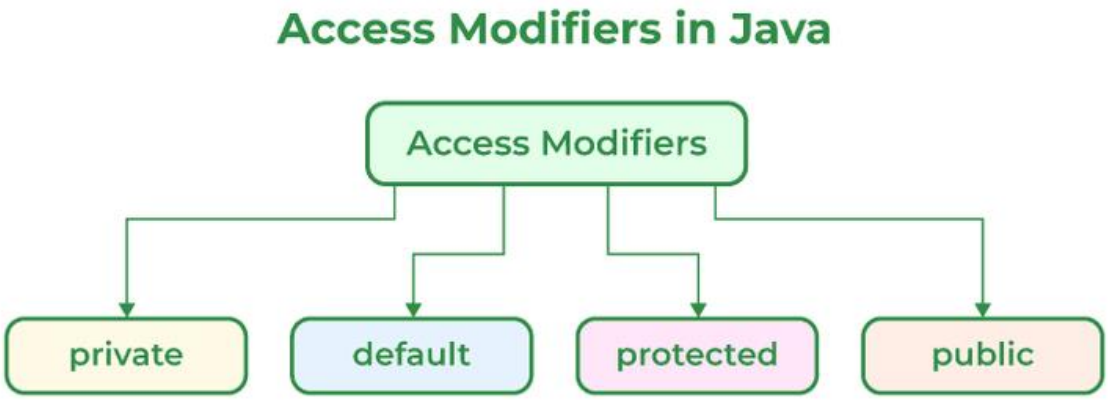
Static(Class) method	Instance method
Static method is associated with a class rather than an object.	The instance method is associated with an object rather than a class.
Static methods can be called using the class name only without creating an instance of a class.	The instance methods can be called on a specific instance of a class using the object reference.
Static methods do not have access to **this** keyword**.**	Instance methods have access to **this** keyword**.**
Static methods can access only static members of the class.	Instance methods can access both static and non-static methods of the class.

64. What is this keyword in Java?



'this' is a keyword used to reference a variable that refers to the current object.

65. What are Brief Access Specifiers and Types of Access Specifiers?



66. What will be the initial value of an object reference which is defined as an instance variable?

The initial value of an object reference which is defined as an instance variable is a NULL value.

67. What is an object?

The object is a real-life entity that has certain properties and methods associated with it. The object is also defined as the instance of a class. An object can be declared using a new keyword.

68. What are the different ways to create objects in Java?

Methods to create objects in Java are mentioned below:

1. Using new keyword
2. Using new instance

3. Using clone() method
4. Using deserialization
5. Using the newInstance() method of the Constructor class

69. What is the constructor?

Constructor is a special method that is used to initialize objects. Constructor is called when a object is created. The name of constructor is same as of the class.

70. What happens if you don't provide a constructor in a class?

If you don't provide a constructor in a class in Java, the compiler automatically generates a default constructor with no arguments and no operation which is a default constructor.

71. How many types of constructors are used in Java?

There are two types of constructors in Java as mentioned below:

1. Default Constructor
2. Parameterized Constructor

72. What is the purpose of a default constructor?

Constructors help to create instances of a class or can be said to create objects of a class. Constructor is called during the initialization of objects. A default constructor is a type of constructor which do not accept any parameter, So whatever value is assigned to properties of the objects are considered default values.

73. What are the differences between the constructors and methods?

Java constructors are used for initializing objects. During creation, constructors are called to set attributes for objects apart from this few basic differences between them are:

1. Constructors are only called when the object is created but other methods can be called multiple times during the life of an object.
2. Constructors do not have a return type, whereas methods have a return type, which can be void or any other type.
3. Constructors are used to setting up the initial state but methods are used to perform specific actions.

74. What is an Interface?

An interface in Java is a collection of static final variables and abstract methods that define the contract or agreement for a set of linked classes. Any class that implements an interface is required to implement a specific set of methods. It specifies the behavior that a class must exhibit but not the specifics of how it should be implemented.

75. Give some features of the Interface.

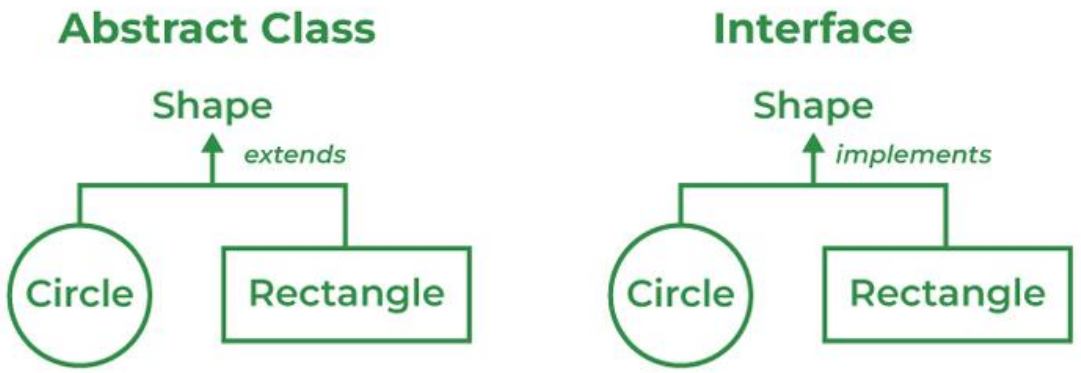
- The interface can help to achieve total abstraction.
- Allows us to use multiple inheritances in Java.

- Any class can implement multiple interfaces even when one class can extend only one class.
- It is also used to achieve loose coupling.

76. What is a marker interface?

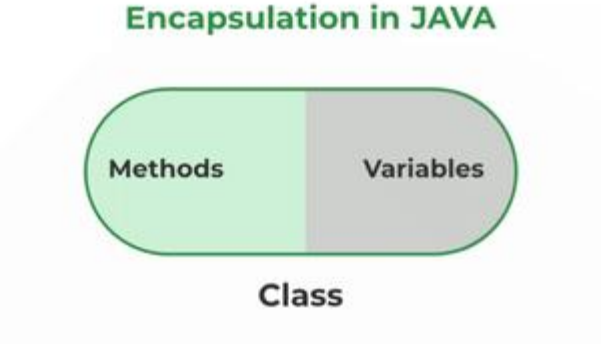
An Interface is recognized as an empty interface (no field or methods) it is called a marker interface. Examples of marker interfaces are Serializable, Cloneable, and Remote interfaces.

77. What are the differences between abstract class and interface?

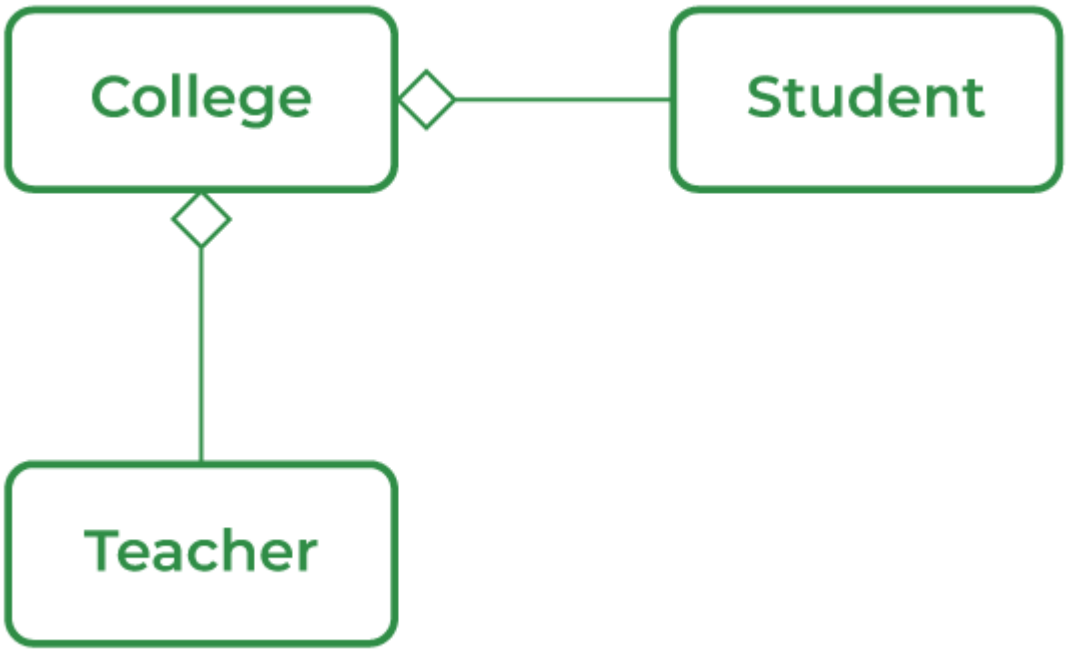


78. What do you mean by data encapsulation?

Basically, it bundles data and methods that operate on that data within a single unit. Encapsulation is achieved by declaring the instance variables of a class as private, which means they can only be accessed within the class.



79. What do you mean by aggregation?



Aggregation is a term related to the relationship between two classes best described as a "has-a" relationship

80. What is the ‘IS-A ‘ relationship in OOPs Java?

'IS-A' is a type of relationship in OOPs Java where one class inherits another class.

81. Define Inheritance.

When an object that belongs to a subclass acquires all the properties and behavior of a parent object that is from the superclass, it is known as inheritance.

82. What are the different types of inheritance in Java?

Inheritance is the method by which the Child class can inherit the features of the Super or Parent class. In Java, Inheritance is of four types:

- **Single Inheritance:** When a child or subclass extends only one superclass, it is known to be single inheritance. Single-parent class properties are passed down to the child class.
- **Multilevel Inheritance:** When a child or subclass extends any other subclass a hierarchy of inheritance is created which is known as multilevel inheritance. In other words, one subclass becomes the parent class of another.
- **Hierarchical Inheritance:** When multiple subclasses derive from the same parent class is known as Hierarchical Inheritance. In other words, a class that has a single parent has many subclasses.
- **Multiple Inheritance:** When a child class inherits from multiple parent classes is known as Multiple Inheritance. In Java, it only supports multiple inheritance of interfaces, not classes.

83. What is multiple inheritance? Is it supported by Java?

A component of the object-oriented notion known as multiple inheritances allows a class to inherit properties from many parent classes. When methods with the same signature are present in both superclasses and subclasses, an issue arises. The method's caller cannot specify to the compiler which class method should be called or even which class method should be given precedence.

84. What is an association?

The association is a relation between two separate classes established through their Objects. It represents Has-A's relationship**.**

85. What do you mean by aggregation?

Aggregation is a relationship between two entities where one entity is associated with another, but they can exist independently. It represents a "has-a" relationship. In Aggregation, the contained object does not get destroyed when the container object is destroyed.

86. What is the composition of Java?

Composition in Java is a design principle where one class contains an instance of another class and establishes a strong relationship between them. The child object cannot exist independently of the parent object.

87. Can the constructor be inherited?

No, we can't inherit a constructor.

88. What is Polymorphism?

An object behaving in multiple forms — via **overloading** (compile-time) or **overriding** (run-time).

89. What is Runtime Polymorphism?

Also called **dynamic method dispatch**; method call is resolved at runtime based on the object type.

90. What is Method Overriding?A subclass provides a new implementation for a superclass method with the **same signature**.

91 What is Method Overloading?

Same method name with different **parameter lists** (type/number). Resolved at **compile-time**.

92. Can we override static methods?

No. Static methods belong to the class, not instances.

97.. Can we override overloaded methods?

Yes. Overloaded methods can be overridden like normal methods.

98. Can we overload the main() method?

Yes. Main method can be overloaded, but JVM calls only main(String[] args).

99. Difference: Overloading vs Overriding

Feature	Overloading	Overriding
Time of Resolution	Compile-time	Runtime
Inheritance Needed	No	Yes
Parameters	Must differ	Must be same
Return Type	Can differ	Must be same (or subtype)
Scope	Same class	Across parent & child

100. Can private methods be overridden?

No. Private methods are not visible to subclasses.

101. Can scope be changed in overridden methods?

Yes, but only wider or same access modifiers are allowed (e.g., protected → public is okay).

102. Can we change throws clause in overriding?

Yes, but only to:

Same exception

Subclass exception

No exception

Cannot throw broader checked exceptions.

103. Does Java support virtual functions?

Yes. All non-static, non-final methods are virtual by default.

104. What is Abstraction?

Hiding internal details and showing only the functionality (e.g., car controls vs engine working).

105. What is an Abstract Class?

A class with abstract methods. Cannot be instantiated. Used to define blueprints.

106. When to use Abstract Methods?

When base class defines method structure, but child class should implement it.

107. How to avoid serialization in child class if parent is Serializable?

Override writeObject() in child and throw NotSerializableException.

COLLECTION

1. What is Collection Framework in Java?

Collections are units of objects in Java. The collection framework is a set of interfaces and classes in Java that are used to represent and manipulate collections of objects in a variety of ways.

2 . Explain various interfaces used in the Collection framework.

Collection framework implements

1. Collection Interface
2. List Interface
3. Set Interface
4. Queue Interface
5. Deque Interface
6. Map Interface

3 . Why do we need a synchronized ArrayList when we have Vectors (which are synchronized) in Java?

ArrayList is in need even when we have Vectors because of certain reasons:

1. ArrayList is faster than Vectors.
2. ArrayList supports multithreading whereas Vectors only supports single-thread use.

3 . Why can’t we create a generic array?

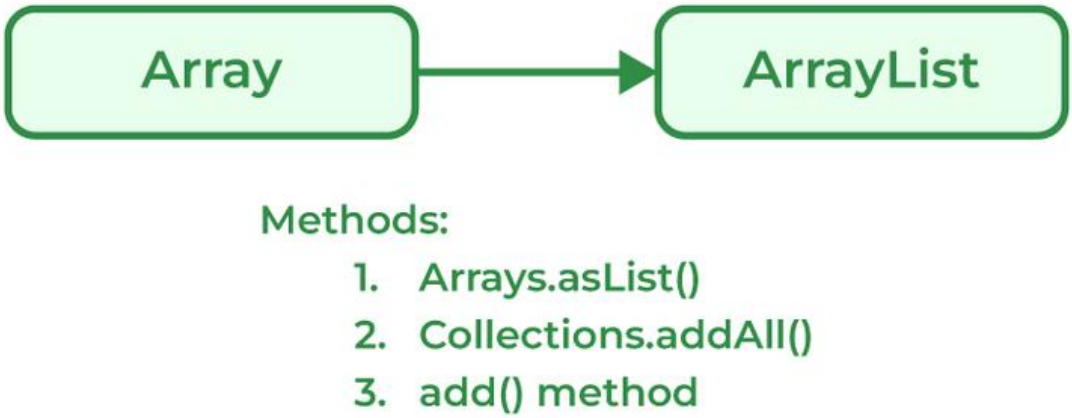
Generic arrays can't be created because an array carries type information of its elements at runtime because of which during runtime it throw 'ArrayStoreException' if the elements' type is not similar. Since generics type information gets erased at compile time by Type Erasure, the array store check would have been passed where it should have failed.

4. Can you explain how elements are stored in memory for both regular arrays and ArrayLists in Java? . Explain.

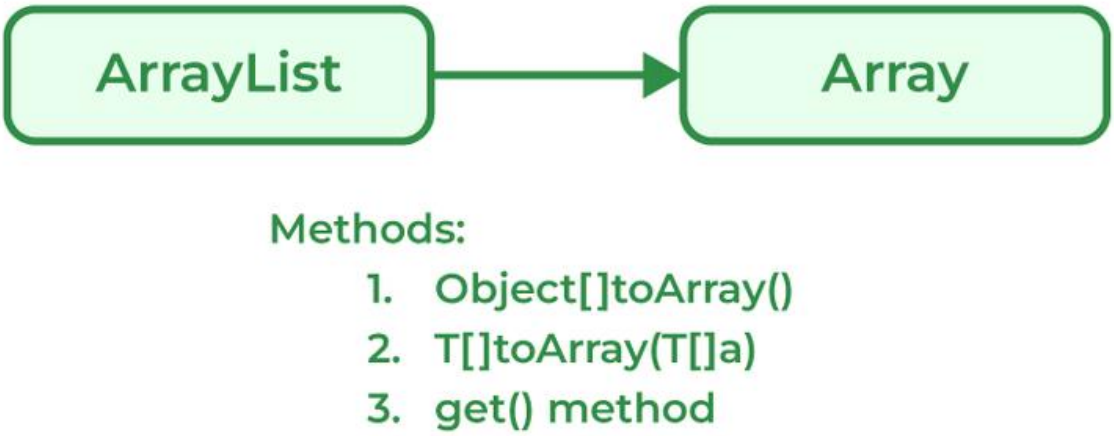
The elements of a regular array in Java are stored in contiguous memory locations, meaning that each element is stored in a sequential block of memory.

the ArrayList class implements a dynamic array, which means that its size can change as elements are added or removed.

6. **Explain the method to convert ArrayList to Array and Array to ArrayList.**



Conversion of ArrayList to Array



7. How to make Java ArrayList Read-Only?

An ArrayList can be made ready only using the method provided by Collections using the Collections.unmodifiableList() method.

8 . What is a Vector in Java?

Vectors in Java are similar and can store multiple elements inside them. Vectors follow certain rules mentioned below:

1. Vector can be imported using Java.util.Vector.
2. Vector is implemented using a dynamic array as the size of the vector increases and decreases depending upon the elements inserted in it.
3. Elements of the Vector using index numbers.

4. Vectors are synchronized in nature means they only used a single thread (only one process is performed at a particular time).
5. The vector contains many methods that are not part of the collections framework.

9 . What is a priority queue in Java?

A priority queue is an abstract data type similar to a regular queue or stack data structure. Elements stored in elements are depending upon the priority defined from low to high. The PriorityQueue is based on the priority heap.

10. Explain the LinkedList class.

LinkedList class is Java that uses a doubly linked list to store elements. It inherits the AbstractList class and implements List and Deque interfaces. Properties of the LinkedList Class are mentioned below:

1. LinkedList classes are non-synchronized.
2. Maintains insertion order.
3. It can be used as a list, stack, or queue.

11. What is the Stack class in Java and what are the various methods provided by it?

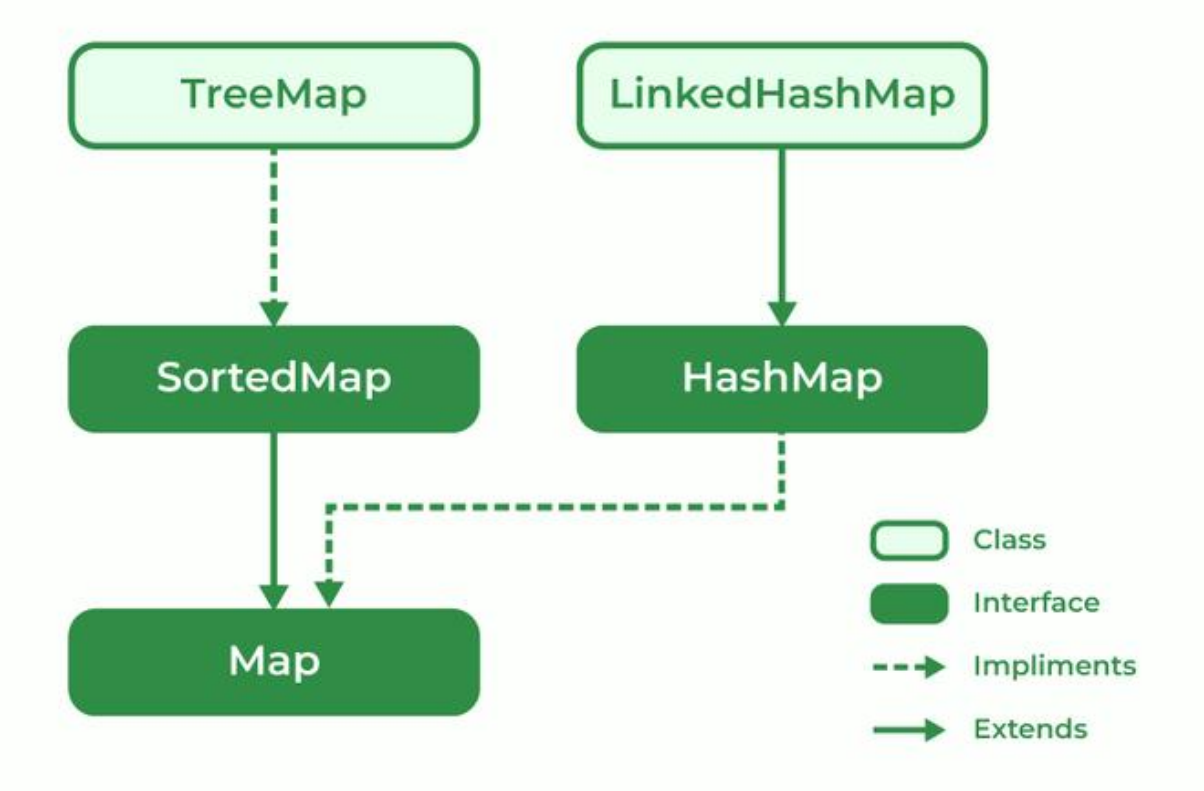
A Stack class in Java is a LIFO data structure that implements the Last In First Out data structure.

- **peek():empty(): push(): pop():search()**

12. What is Set in the Java Collections framework and list down its various implementations?

- Sets are collections that don't store duplicate elements
  HashSet: HashSet in Java, stores the elements in a has table which provides faster lookups and faster insertion. HashSet is not ordered.
- **LinkedHashSet:** LinkedHashSet is an implementation of HashSet which maintains the insertion order of the elements.
- **TreeSet:** TreeSet stores the elements in a sorted order that is determined by the natural ordering of the elements or by a custom comparator provided at the time of creation.

13. What is a Map interface in Java?



The map interface is present in the Java collection and can be used with Java.util package. A map interface is used for mapping values in the form of a key-value form. The map contains all unique keys. Also, it provides methods associated with it like containsKey(), contains value (), etc.

There are multiple types of maps in the map interface as mentioned below:

1. SortedMap
2. TreeMap
3. HashMap
4. LinkedHashMap

14. Explain Treemap in Java

TreeMap is a type of map that stores data in the form of key-value pair. It is implemented using the red-black tree. Features of TreeMap are :

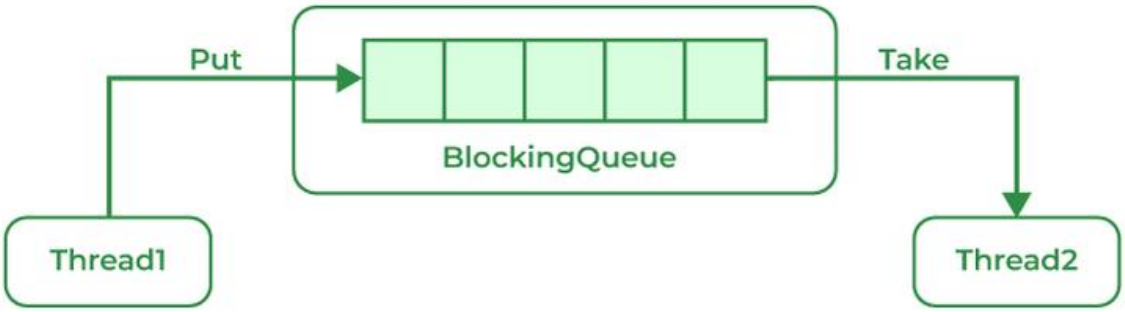
1. It contains only unique elements.
2. It cannot have a NULL key
3. It can have multiple NULL values.
4. It is non-synchronized.
5. It maintains ascending order.

15. What is EnumSet?

EnumSet is a specialized implementation of the Set interface for use with enumeration type. A few features of EnumSet are:

1. It is non-synchronized.
2. Faster than HashSet.
3. All of the elements in an EnumSet must come from a single enumeration type.
4. It doesn't allow null Objects and throws NullPointerException for exceptions.
5. It uses a fail-safe iterator.

16. What is BlockingQueue?



A blocking queue is a Queue that supports the operations that wait for the queue to become non-empty while retrieving and removing the element, and wait for space to become available in the queue while adding the element.

17. What is the ConcurrentHashMap in Java and do you implement it?

ConcurrentHashMap is implemented using Hashtable.

18. What is an enumeration?

Enumeration is a user-defined data type. It is mainly used to assign names to integral constants, the names make a program easy to read and maintain. The main objective of the enum is to define user-defined data types.

Example:

```
// A simple enum example where enum is declared

// outside any class (Note enum keyword instead of

// class keyword)

enum Color

{

    RED, GREEN, BLUE;

}
```

21. What is the difference between Collection and Collections?

Collection	Collections
The Collection is an Interface.	Collections is a class.
It provides the standard functionality of data structure.	It is to sort and synchronize the collection elements.
It provides the methods that can be used for the data structure.	It provides static methods that can be used for various operations.

. Differentiate between Array and ArrayList in Java.

Array	ArrayList
Single-dimensional or multidimensional	Single-dimensional
For and for each used for iteration	Here iterator is used to traverse riverArrayList
length keyword returns the size of the array.	size() method is used to compute the size of ArrayList.
The array has Fixed-size.	ArrayList size is dynamic and can be increased or decreased in size when required.
It is faster as above we see it of fixed size	It is relatively slower because of its dynamic nature

ArrayList	LinkedList
ArrayList is Implemented as an expandable Array.	LinkedList is Implemented as a doubly-linked list.
In ArrayList, Elements are stored in contiguous memory locations	LinkedList Elements are stored in non-contiguous memory locations as each element has a reference to the next and previous elements.
ArrayLists are faster for random access.	LinkedLists are faster for insertion and deletion operations
ArrayLists are more memoy efficient.	LinkedList is less memory efficient
ArrayList	Vector
ArrayList is implemented as a resizable array.	Vector is implemented as a synchronized, resizable array.
ArrayList is not synchronized.	The vector is synchronized.
ArrayLists are Faster for non-concurrent operations.	Vector is Slower for non-concurrent operations due to added overhead of synchronization.

Iterator	ListIterator
Can traverse elements present in Collection only in the forward direction.	Can traverse elements present in Collection both in forward and backward directions.

Iterator	ListIterator
Used to traverse Map, List, and Set.	Can only traverse List and not the other two.
Indexes can't be obtained using Iterator	It has methods like nextIndex() and previousIndex() to obtain indexes of elements at any time while traversing the List.

HashMap	HashTable
HashMap is not synchronized	HashTable is synchronized
One key can be a NULL value	NULL values not allowed
The iterator is used to traverse HashMap.	Both Iterator and Enumertar can be used
HashMap is faster.	HashTable is slower as compared to HashMap.

Iterator	Enumeration
The Iterator can traverse both legacies as well as non-legacy elements.	Enumeration can traverse only legacy elements.

Iterator	Enumeration
The Iterator is fail-fast.	Enumeration is not fail-fast.
The Iterators are slower.	Enumeration is faster.
The Iterator can perform a remove operation while traversing the collection.	The Enumeration can perform only traverse operations on the collection.

Comparable	Comparator
The interface is present in java.lang package.	The Interface is present in java.util package.
Provides compareTo() method to sort elements.	Provides compare() method to sort elements.
It provides single sorting sequences.	It provides multiple sorting sequences.
The logic of sorting must be in the same class whose object you are going to sort.	The logic of sorting should be in a separate class to write different sorting based on different attributes of objects.
Method sorts the data according to fixed sorting order.	Method sorts the data according to the customized sorting order.

Set	Map
The Set interface is implemented using java.util package.	The map is implemented using java.util package.
It can extend the collection interface.	It does not extend the collection interface.
It does not allow duplicate values.	It allows duplicate values.
The set can sort only one null value.	The map can sort multiple null values.

EXCEPTION HANDLING

1. What is Exception Handling?

An exception is an event that interrupts the normal execution flow of a program.

Java provides a built-in mechanism to handle such events: Exception Handling.

2. How many types of exceptions can occur in a Java program?

Built-in Exceptions:

User-Defined Exceptions

3. Difference between an Error and an Exception.

Recovering from Errors is not possible.	**Recover from exceptions by either using a try-catch block or throwing exceptions back to the caller.**
Errors are all unchecked types in Java.	**It includes both checked as well as unchecked types that occur**

4.Explain Runtime Exceptions.

- NullPointerException:
- ArrayIndexOutOfBoundsException:
- ArithmeticException:
- IllegalArgumentException:

5 .What will happen if you put System.exit(0) on the try or catch block? Will finally block execute?

System.exit(int) has the capability to throw SecurityException

6.What purpose do the keywords final, finally, and finalize fulfill?

i). final:

final is a keyword is used with the variable, method, or class so that they can't be overridden.

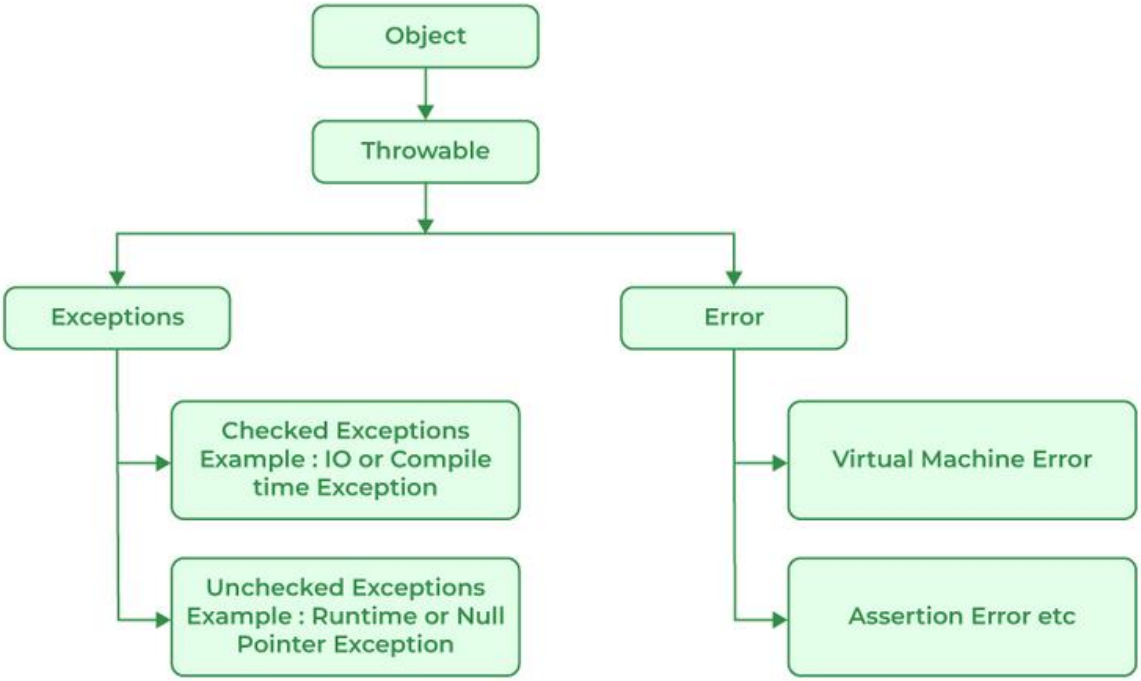
ii). finally

finally is a block of code used with "try-catch" in exception handling. Code written in finally block runs despite the fact exception is thrown or not.

iii). finalize

It is a method that is called just before deleting/destructing the objects which are eligible for Garbage collection to perform clean-up activity.

7 .Explain the hierarchy of Java Exception classes.



MULTITHREADING

1. What is Multitasking?

Executing multiple tasks concurrently.

Achieved in Java using threads.

Example: Creating two threads to print numbers simultaneously.

2. What is a Multithreaded Program?

A program that uses multiple threads running concurrently

Improves:CPU utilization

Application performanc

Responsiveness

3. Advantages of Multithreading

Better responsiveness in user interfaces

Resource sharing among threads

Improved economy by sharing memory

Scalability on multi-core CPU

Efficient communication via synchronization

Optimized system resource usage

Utilization of multiprocessor architecture

4. Two Ways to Create Threads in Java

a) Extending Thread class:

```
class MyThread extends Thread {
    public void run() {
        // thread code
    }}
```

b) Implementing Runnable interface:

```
class MyRunnable implements Runnable {
    public void run() {
    }
}
```

5. What is a Thread?

A lightweight subprocess within a program.

Has its own:

Stack

Program Counter

Local variables

Shares memory with other threads in the same process.

One thread's failure does not affect others.

6. Process vs Thread

Aspect	Process	Thread
Definition	Independent	Part of a process
Weight	Heavyweight	Lightweight
Context Switching	Slower	Faster
Memory Sharing	No	Yes
Address Space	Separate	Common

7. Thread Lifecycle States

New – Thread created but not started

Runnable – Ready to run or running

Blocked – Waiting for a resource

Waiting – Waiting indefinitely for another thread

Terminated – Execution finished or stopped

8. suspend() Method (Deprecated)

Temporarily pauses thread execution.

Alternatives recommended: sleep(), wait(), etc.

```
thread.suspend();  // Suspend the thread
thread.resume();   // Resume execution
```

9. Main Thread

The thread that runs the main() method.

Automatically created at program startup.

Acts as the parent thread to all others.

10. What is a Daemon Thread?

A background thread for tasks like:

Garbage Collection

Event dispatching

Listeners

Ends when all user threads terminate.

```
thread.setDaemon(true);
```

11. Ways Threads Enter Waiting State

sleep() – pauses thread for a fixed time

wait() – waits until notified by another thread

join() – waits for another thread to finish

I/O Operations – waiting for input/output completion

Synchronization blocks – waiting for a lock

12. Multithreading on Single CPU

Achieved using time-slicing (time-sharing).

The OS switches rapidly between threads to simulate concurrency.

Java uses synchronization to prevent race conditions.

13. Thread Priorities in Java

Priority range: 1 to 10

MIN_PRIORITY = 1

NORM_PRIORITY = 5 (Default)

MAX_PRIORITY = 10

thread.setPriority(Thread.MIN_PRIORITY);

14. Why Garbage Collection is Necessary?

Frees up unused memory.

Prevents memory leaks.

Managed by JVM, not manually.

Improves performance and resource efficiency.

15. Drawbacks of Garbage Collection

Causes application pauses

Non-deterministic: timing can't be predicted

May increase memory use (short-lived objects created/discarded rapidly)

16. Types of Garbage Collection

Type Description

Minor GC Reclaims memory in young generation (Eden space)

Major GC Reclaims memory in old generation (tenured space)

Full GC Cleans both young + old generation, may cause a pause

18. What is a Memory Leak in Java?

Occurs when unused objects are not garbage collected due to active references.

Consequences:

Wasted memory

Performance degradation

May cause OutOfMemoryError

Prevent by:

Releasing unused references

Closing I/O/resources properly